Table of Contents

# JupiterOne Data Security

This document describes in detail the data JupiterOne ingests and how your data is protected on our platform.

## Data Protection

### Encryption

Data is fully encrypted both at rest and in transit. This includes all of your account and user data, as well as operational data imported/ingested into the JupiterOne platform.

**Data in Transit** is encrypted via TLSv1.2 or later, using SHA-256 with 2048-bit RSA Encryption or equivalent strength cypher.

*Production Domains:* \*.apps.us.jupiterone.io is the associated production URL that the SSL/TLS certificate has been issued to.

**Data at Rest** is hosted in our production AWS environments, using the managed RDS/Neptune, DynamoDB, and S3 services. All database instances, tables, and S3 buckets with customer data have server-side encryption enabled, using AWS KMS for key management. KMS encryption keys are scheduled to rotate annually.

In addition to encryption, managed backup is enabled for the database clusters. For S3 buckets, cross-account replication is enabled to back up data to a different region for disaster recovery. All backup data is fully encrypted the same way as its source.

### Multi-tenancy

JupiterOne is a multi-tenancy, software-as-a-service platform hosted in AWS. Customer data is logically partitioned/segregated by software via a unique accountId associated with every piece of data. Access to data is restricted to within each unique account for users granted proper access to that account. This is a standard pattern used by cloud infrastructure and SaaS providers.

## External Data Ingestion/Import

High level information of the JupiterOne Data Model can be found [here](https://support.jupiterone.io/hc/en-us/articles/360011556113-JupiterOne-Data-Model).

For more details on data ingested for each managed integration, see their corresponding documentation:

* [AWS](https://support.jupiterone.io/hc/en-us/articles/360012626754-AWS-Data-and-Integration-Details)
* [Bitbucket](https://support.jupiterone.io/hc/en-us/articles/360012758713-Bitbucket-Data-and-Integration-Details)
* [Github](https://support.jupiterone.io/hc/en-us/articles/360012627134-Github-Data-and-Integration-Details)
* Okta

### Access Permissions Needed to Integrated Environments

Access to your environments is needed in order to ingest data, or to enable workflow automation (future capability).

In general, JupiterOne only requires read-only, security-auditor-type access permissions to your environments. Additionally, this read-only access only applies to configurations and meta data, not the actual data content. For example, we do **NOT** read S3 objects data from a connected AWS account, or the actual source code of a connected Bitbucket/Github account.

Additional level of access may be needed for workflow automation. For example, integration with Jira to automatically create an issue when a new Vulnerability finding is added; or to post to a Slack channel/user to send a security alert notification.

You are always in control of the actual permissions granted for each integration. More details of the access permissions required for each managed integration can be found in its corresponding documentation listed above.

### Custom Data Import

Additionally, JupiterOne supports the ability for you to add custom data by

* Manually adding entities via Web UI in the Asset Inventory app;
* Adding bulk number of entities via CSV import; or
* Adding custom entities via custom integrations using the public API.

## Data Ownership and Access

You retain full ownership of all data that is ingested via integrations, API or manual importing/creation. Data is stored in JupiterOne’s production environment in AWS, protected via encryption and replication as specified in the first section.

### Infrastructure and Operational Access

JupiterOne infrastructure is built on a **Zero Trust** security model, where access to production is *highly restricted*.

The production environment is virtually “air-gapped” such that there is no SSH, “bastion host”, or VPN connectivity into the production systems to prevent unintended network access to databases and other production servers. We do not allow internal access to production data by any JupiterOne team member. All necessary operational support and maintenance jobs are performed via automation where the automation code is fully documented, reviewed, and approved, ensuring end-to-end traceability.

Our production environment incorporates multiple layers of security monitoring, using JupiterOne itself as well as third party security solutions. Additionally, our software development includes rigorous code analysis and continuous testing practices to ensure we proactively identify any security vulnerability. Our infrastructure-as-code operational model and automated change management process allows us to deploy security patches within minutes of identification and remediation of an issue.

You can review our published [security model](https://security.lifeomic.com/psp/model/) and corresponding [policies and procedures](https://security.lifeomic.com/psp/) for more details on our operational, infrastructure, and software development security.

## Application Access

Access to the JupiterOne application and your accounts/data on the platform is enabled over HTTPS, through either the JupiterOne web apps or the public APIs.

*Note:* \*.us.jupiterone.io is the current production domain.

### User Logins

Each user has a unique user login to the JupiterOne platform and apps. Users may be invited to one or multiple organizational accounts on JupiterOne.

#### Password Policy

Users are required to select a strong password meeting the following password policy requirements in order to create a login and authenticate to the system:

* Minimum of 8 characters
* Must contain an uppercase letter
* Must contain a lowercase letter
* Must contain a number
* Must contain a special character

#### Single Sign On (SSO)

JupiterOne currently supports single sign on (SSO) via:

* Google
* SAML

#### Multi-Factor Authentication (MFA) / Two-Step Verification (2SV)

Multi-Factor Authentication (MFA) or Two-Step Verification (2SV) is strongly recommended for all users on the JupiterOne platform. This needs to be enabled and configured via your SSO provider (Google or your SAML IdP such as Okta or OneLogin).

### Access Control

In order to support potential complex access control use cases, JupiterOne platform implements Attribute Based Access Control (ABAC).

A good general overview of ABAC is sections 1 and 2 of NIST’s [Guide to Attribute Based Access Control](https://csrc.nist.gov/publications/detail/sp/800-162/final). The absolute basics of ABAC are that you have a subject (e.g. a user) who wants to perform some operation ( e.g. download) on an object (e.g. a file) in some environment. The subject, object and environment all have attributes (i.e. key/value pairs), and there are policies that control the privileges (i.e. what operations the subject can perform) given the attributes.

Access policies defined in JupiterOne are associated with a **User Group** and **Users** are invited/added as members to one or more groups.

* A *Read-Only* access policy is predefined and associated with the default **Users** group.
* A *Full-Access* policy is also predefined and associated with the default **Administrators** group.
* The ability to customize and add granular access control policies is to be released in 1Q2019.

### API Access

JupiterOne API is available at: https://api.us.jupiterone.io/

We use [OAuth 2.0](https://oauth.net/2/) for authorization, which means in order to access data a user must authenticate and the requesting app must be authorized. Implicit grant, authorization code, and client credentials flows are supported. Authorization code is recommended for web apps, which involves utilizing both the authorize and token API resources. When using the authorization code grant flow, it is also recommended to use Proof Key for Code Exchange (PCKE) to mitigate authorization code intercept attacks. Contact us if building a native app which can securely perform client credentials flow.

Additionally, each user on the platform can create an API key that can be passed along with request to act on behalf of that user.

*Note: the UI for self-service configuration of OAuth and user API key is targeted to be available in 1Q2019.*

### Support Access to Your JupiterOne Account(s)

A JupiterOne Support User is by default added to a new account during free trial, proof-of-concept evaluation, or initial account onboarding. This is to facilitate better support and training to using the platform.

* The support user’s login can either be the individual Security Engineer/Architect designated to your account (e.g. firstname.lastname@jupiterone.io) or the general support login (i.e. callisto@jupiterone.io).
* The support user can be removed by an account administrator at any time, should you determine that ongoing regular support is no longer needed.
* You have the option and administrative privilege to add the support user back at any time, when support is needed in the future.